```
1
2   from a0_items import *
3
4
5
6   #create db file  (./ = same folder, ../ = outside current folder)
7   # https://stackoverflow.com/questions/5801170/python-sqlite-create-table-if-not-exists-problem
8   def sqlite_create_file (sqlite_path, sql):
9
10      if os.path.isfile(sqlite_path):
11
12          print (sqlite_path + ' alraedy exists')
13
14      else:
15
16          conn = sqlite3.connect (sqlite_path)
17
18          c = conn.cursor()
19          c.execute(sql)
20
21          conn.commit()
22
23          print (sqlite_path + ' NOT exists, is created!')
24
25
26  '''
27  db_file = 'DB_shane.db'
28  sql_create = 'create table if not exists HELLO_table (ID integer primary key autoincrement, name text)'
29
30  sqlite_create_file (sqlite_path = 'DB_shane.db', sql = sql_create)
31  '''
32
33
34  # Use loop to insert multiple values of ONE column only
35  def sqlite_insert_data (sqlite_path, sql_insert, data):
36
37      conn = sqlite3.connect(sqlite_path)
38      c = conn.cursor()
39
40      c.execute(sql_insert,(data,)) #must have , at the end of data variable, 1 element tuple -- https://stackoverflow.com/questions/16856647/sqlite3-programmingerror-incorrect-number-of-bindings-supplied-the-current-sta
41
42      conn.commit()
43
44  '''
45  sql = 'INSERT INTO Country_language (country_code)VALUES(?)'
46  sqlite_insert_data (sqlite_path='../../data/cooked/DB_country_language/China.db', sql_insert=sql, data='just_country_code')
47  '''
48
49
50
51  # Use loop to insert multiple values of ONE column only
52  def sqlite_insert_data_multi_columns (sqlite_path, sql, data):
53
54      conn = sqlite3.connect(sqlite_path)
55      c = conn.cursor()
56
57      c.execute(sql, data) # https://pynative.com/python-sqlite-insert-into-table/
58      conn.commit()
59
60
```

```
61
62  def test_code ():
63
64      sqlite_insert_with_param = """INSERT INTO SqliteDb_developers
65                            (id, name, email, joining_date, salary)
66                            VALUES (?, ?, ?, ?, ?);"""
67
68      data_tuple = ('2', 'TFM', 'hello@email.com')
69
70      sqlite_insert_data_multi_columns (sqlite_path ='sqlite.db', sql =
        sqlite_insert_with_param, data = data_tuple)
71
72
73
74  ##########################################################################################
    ##############
75  ##########################################################################################
    ##############
76
77  # https://pynative.com/python-sqlite-update-table/
78  # Use loop to update multiple values. But update 1 column at a time (ref_column =
    reference column)
79
80  def sqlite_update_data (sqlite_path, sql_update, update_data):
81
82      try:
83
84          sqliteConnection = sqlite3.connect(sqlite_path)
85          cursor = sqliteConnection.cursor()
86          print("Connected to SQLite")
87
88          cursor.execute(sql_update, update_data)
89          sqliteConnection.commit()
90          print("Record Updated successfully")
91          cursor.close()
92
93      except sqlite3.Error as error:
94
95          print("Failed to update sqlite table", error)
96
97
98  '''
99  sql = "Update Country_language set country_code = ? where id = ?"
100 data_tuple = ('TEST123', '1')
101 sqlite_update_data (sqlite_path='../../data/cooked/DB_country_language/China.db',
    sql_update=sql, update_data = data_tuple)
102 '''
103
104
105
106 def sqlite_delete_table (sqlite_path, table_name):
107
108     conn = sqlite3.connect(Sqlite_file)
109     c = conn.cursor()
110
111     sql = 'drop table ' + table_name
112     c.execute(sql)
113
114     conn.commit()
115
116
117
118 def sqlite_select_ALL_data (sqlite_path, sql):
119
120     LIST_sqlite_data = []
121
122     conn = sqlite3.connect(sqlite_path)
```

```
    c = conn.cursor()

    c.execute(sql)

    LIST_sqlite_data = c.fetchall()

    return LIST_sqlite_data

'''
sql_select = "SELECT * FROM Country_language "
LIST_test = sqlite_select_ALL_data (sqlite_path=
'../../data/cooked/DB_country_language/China.db', sql=sql_select)

print (len(LIST_test))

for x in LIST_test:

    if len(str(x))==0 or x is None or x == None :

        print (" None ")

    else:

        print (" NOT none")
'''



def sqlite_delete_ALL_data (sqlite_path, table_name):

    conn = sqlite3.connect(sqlite_path)
    c = conn.cursor()

    sql = 'DELETE FROM ' + table_name
    cur = conn.cursor()
    cur.execute(sql)
    conn.commit()


#sqlite_delete_ALL_data (sqlite_path= '../../data/cooked/DB_country_language/China.db',
table_name='Country_language')
```